NASA/TM–2000–210104

Analytical and Computational Aspects of Collaborative Optimization

Natalia M. Alexandrov
Langley Research Center, Hampton, Virginia

Robert Michael Lewis
Institute for Computer Applications in Science and Engineering
Langley Research Center, Hampton, Virginia

April 2000

NASA/TM–2000–210104

Analytical and Computational Aspects of Collaborative Optimization

Natalia M. Alexandrov
Langley Research Center, Hampton, Virginia

Robert Michael Lewis
Institute for Computer Applications in Science and Engineering
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681–2199

April 2000

Available from:

NASA Center for AeroSpace Information (CASI)
7121 Standard Drive
Hanover, MD 21076–1320
(301) 621–0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161–2171
(703) 605–6000

ANALYTICAL AND COMPUTATIONAL ASPECTS OF COLLABORATIVE OPTIMIZATION

NATALIA M. ALEXANDROV[*] AND ROBERT MICHAEL LEWIS[†]

Abstract. Bilevel problem formulations have received considerable attention as an approach to multidisciplinary optimization in engineering. We examine the analytical and computational properties of one such approach, collaborative optimization. The resulting system-level optimization problems suffer from inherent computational difficulties due to the bilevel nature of the method. Most notably, it is impossible to characterize and hence identify solutions of the system-level problems because the standard first-order conditions for solutions of constrained optimization problems do not hold. The analytical features of the system-level problem make it difficult to apply conventional nonlinear programming algorithms. Simple examples illustrate the analysis and the algorithmic consequences for optimization methods. We conclude with additional observations on the practical implications of the analytical and computational properties of collaborative optimization.

Key words. Bilevel optimization, collaborative optimization, constraint qualification, decomposition, multidisciplinary design optimization, multilevel optimization, nonlinear programming, optimality conditions

1. Introduction.

Multidisciplinary design optimization, or MDO, is concerned with systematic approaches to the design optimization of complex, coupled engineering systems, where "multidisciplinary" refers to the different aspects that must be included in a design problem. The design process is extremely complex because engineering systems are governed by the considerations of all the contributing disciplines. The design of aerospace vehicles involves, for instance, aerodynamics, structural analysis, propulsion, and control, among many other disciplines. For the purposes of this paper, we mean by MDO a subset of the total design problem—probably in the conceptual or preliminary phases—that can be formulated as a mathematical optimization problem (nonlinear program).

We examine here a class of bilevel approaches to MDO that has recently received attention under the name of collaborative optimization (e.g., [11–13, 23, 28]). The fundamental idea also appeared previously in [1, 6, 7, 25–27, 33, 37]. Collaborative optimization (CO) is an approach to MDO problems based on the decomposition of the problem along the lines of the constituent disciplines. CO seeks to state and solve MDO problems in a way that preserves the autonomy of the disciplinary calculations by eliminating from the system-level problem all those design variables local to individual disciplinary subsystems.

CO is distinguished by its bilevel structure, where consistency among the disciplinary subsystems is enforced via equality constraints in a system-level problem that coordinates the interdisciplinary coupling while trying to improve the system-level performance objective. The values of these constraints are obtained by solving distributed, lower-level optimization subproblems, whose objectives minimize the interdisciplinary inconsistency, subject to satisfying the disciplinary design constraints. (An alternative bilevel approach, developed in [31, 32, 34, 35], maintains interdisciplinary consistency at the system level while seeking to minimize the violation of the disciplinary design constraints at the subsystem level.)

Computational difficulties with the fundamental approach in CO were already observed in [36] and

[*] NASA Langley Research Center, Mail Stop 159, Hampton, Virginia, 23681-2199, n.alexandrov@larc.nasa.gov.

[†] ICASE, Mail Stop 132C, NASA Langley Research Center, Hampton, Virginia, 23681-2199, buckaroo@icase.edu. This research was supported by the National Aeronautics and Space Administration under NASA Contract NAS1-97046.

further confirmed in later numerical tests [2, 22]. This numerical evidence motivated the current study.

As we discuss, collaborative optimization possesses analytical features that lead to serious computational difficulties when conventional nonlinear programming algorithms are applied to the solution of the resulting system-level problem. In particular,

- CO leads to system-level optimization problems that necessarily fail to satisfy the standard Karush-Kuhn-Tucker conditions, either because Lagrange multipliers do not exist, or because the constraint Jacobian is discontinuous at solutions.
- CO formulations lead to system-level problems that are more nonlinear than the original problem.

These features make it very difficult for conventional optimization algorithms to solve the CO system-level problems reliably or efficiently. We believe that these analytical properties of CO are the cause of the numerical difficulties reported in [2, 22, 36].

The computational difficulties reported in connection with CO are an intrinsic property of the bilevel approach used to enforce interdisciplinary consistency via system-level equality constraints, while attempting to minimize the inconsistency via lower-level optimization problems. Because this bilevel approach is so frequently encountered in the multidisciplinary optimization literature, a careful analysis is constructive in clarifying some of the important practical computational issues that arise and which remain to be addressed.

One of the broader points of this paper is that problem formulation has profound, practical algorithmic consequences. Reformulating a perfectly well-posed MDO problem may have pernicious effects on our ability to solve the problem. The transformation to CO destroys the computational character of the original optimization problem, and the resulting problem may be impossible to solve reliably with standard algorithms. The difference in computational behavior of two formulations of the same problem shows that the study of feasibility, optimality, and sensitivity to perturbations is a crucial component in comparing different approaches to formulating MDO problems, as is the study of the practical algorithmic consequences of choosing a specific problem formulation. The general considerations of formulation equivalence are addressed in [4, 5].

We give mathematical statements of the analytical results, which we illustrate with a number of simple examples. The analysis of CO given here is by no means exhaustive. We address only some of the analytical and computational features of immediate import to optimization algorithms.

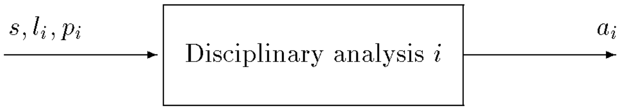
Section 2 describes the two-discipline model problem used to illustrate the properties of CO and its standard formulation as an optimization problem. Section 3 introduces two variants of CO applied to the model problem. Section 4 describes two simple examples used throughout the paper to illustrate the salient characteristics of CO. Section 5 gives the motivation for CO in preparation for Section 6, which contains the analysis of the important analytical features of CO, illustrated with computational examples. Section 7 discusses relaxation of CO designed in order to improve its numerical behavior. Section 8 concludes with additional observations on the practical computational considerations arising in connection with collaborative optimization.

2. A two-discipline model problem.

For clarity, we present our discussion of CO for an MDO problem with two disciplines: Discipline 1 and Discipline 2. For example, they might represent the aeroelastic interaction between aerodynamics (Discipline 1) and structural analysis (Discipline 2) for a wing in steady-state flow.

2.1. Problem components.

The disciplinary subsystems are the building blocks of MDO problems. We assume that each discipline is based on a disciplinary analysis depicted schematically as the input-output relation in the following figure:



Each discipline takes as its input a set of design variables (s, l_i) and parameters p_i. Each discipline produces a set of analysis outputs a_i. We use a_i to represent the totality of outputs from a given discipline, including all data that are passed to the other discipline as parameters, but also, perhaps, quantities passed to design constraints and objectives. The *system-level design variables* s are those *shared* by both disciplines. The *disciplinary design variables* l_1 and l_2 are *local* to Disciplines 1 and 2, respectively. Parameters p_i are derived from the analysis outputs a_j, $j \neq i$, of the other discipline. They are not directly manipulated by the designer in Discipline i. For instance, in our aeroelastic example, the input p_1 from structures to aerodynamics would include the wing shape, while the input p_2 from aerodynamics to structures would include the aerodynamic loads.

The disciplinary input-output relations have the functional form

$$a_i = A_i(s, l_i, p_i).$$

The disciplinary analyses A_1 and A_2 are assumed to be independently soluble. That is, for Discipline 1 we assume that, given appropriate values of inputs (s, l_1, p_1) to Discipline 1, we can compute the disciplinary output a_1 via the Discipline 1 analysis

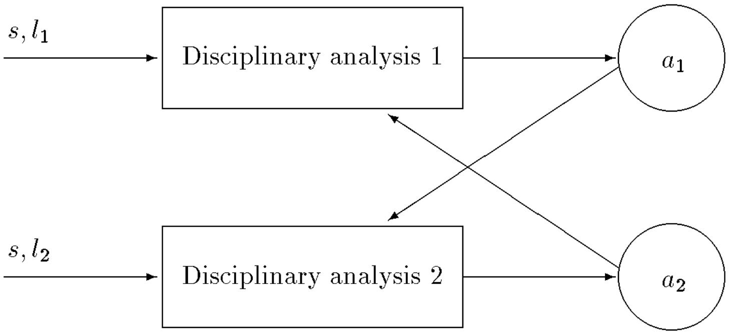
$$a_1 = A_1(s, l_1, p_1).$$

(By "appropriate" we mean input values for which the analysis is defined.) Continuing with our aeroelastic illustration, given values p_1 for the shape of the wing, we can compute the flow a_1 around it. Likewise, given appropriate values of inputs (s, l_2, p_2) to Discipline 2, we assume we can compute the disciplinary outputs a_2 as

$$a_2 = A_2(s, l_2, p_2).$$

In our example, given values of the aerodynamic loads p_2, we can compute the structural response a_2.

2.2. Multidisciplinary analysis. In the context of the MDO problem, the coupled multidisciplinary analysis system (MDA) reflects the physical requirement that a solution simultaneously satisfy the two disciplinary analyses. The input parameters p_i to each discipline are now required to correspond to some (or all) of the outputs a_j from the other disciplinary analysis. Schematically, we have



We write the consistent multidisciplinary analysis system as a simultaneous system of equations. Given (s, l_1, l_2), we have

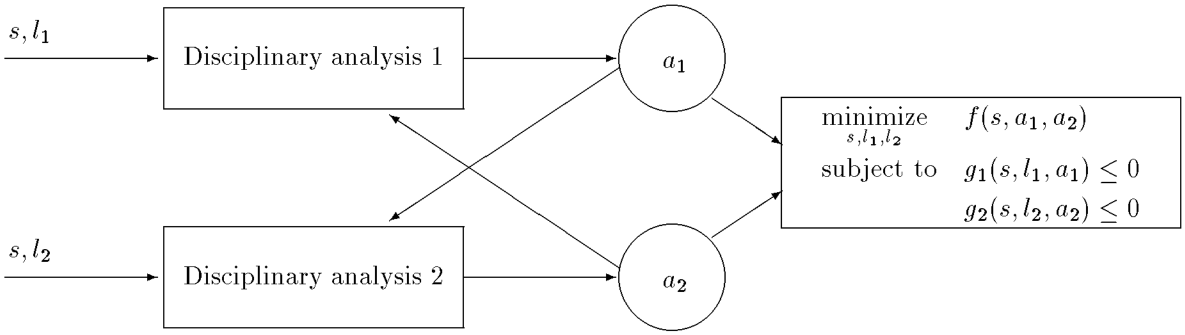
$$a_1 = A_1(s, l_1, a_2) \tag{2.1}$$
$$a_2 = A_2(s, l_2, a_1), \tag{2.2}$$

where solving the first equation results in the analysis outputs a_1 of Discipline 1, and solving the second equation produces the analysis outputs a_2 of Discipline 2. The multidisciplinary analysis thus implicitly defines a_1 and a_2 as functions of (s, l_1, l_2):

$$a_1 = a_1(s, l_1, l_2), \qquad a_2 = a_2(s, l_1, l_2).$$

Solving the coupled equations (2.1)–(2.2) leads to a full multidisciplinary analysis, in which the coupled disciplines give a physically consistent (and thus meaningful) result. The disciplinary responses a_i describe part of the behavior of the system. Again, if Discipline 1 represents aerodynamic analysis of the flow around a wing and Discipline 2 represents structural analysis of the wing, a_1 and a_2 may represent the flow field near the wing and the deformed shape of the wing due to structural response and aerodynamic loads, respectively. The calculation of the flow field a_1 requires the shape of the wing, which is contained in a_2, while the calculation of the wing deformation a_2 requires the aerodynamic loads, contained in a_1.

2.3. A standard formulation of the two-discipline MDO problem. We now turn to coupling the two disciplines in connection with a design optimization problem. Given the need to satisfy MDA at a solution, the most natural optimization problem formulation, arguably, is to impose an optimizer over MDA. In fact, this approach has been commonplace in engineering for many years. We will use this standard formulation to represent the original problem, i.e., the problem one ideally wishes to solve. The flow of information in this formulation is depicted as follows:


s, l_1
Disciplinary analysis 1
a_1
s, l_2
Disciplinary analysis 2
a_2
$\underset{s,l_1,l_2}{\text{minimize}}$ $f(s, a_1, a_2)$
subject to $g_1(s, l_1, a_1) \leq 0$
$g_2(s, l_2, a_2) \leq 0$

The mathematical statement of the standard MDO formulation is

$$\begin{array}{ll} \underset{s,l_1,l_2}{\text{minimize}} & f(s, a_1(s, l_1, l_2), a_2(s, l_1, l_2)) \\ \text{subject to} & g_1(s, l_1, a_1(s, l_1, l_2)) \geq 0 \\ & g_2(s, l_2, a_2(s, l_1, l_2)) \geq 0, \end{array} \tag{2.3}$$

where, given (s, l_1, l_2), we solve the multidisciplinary analysis system (2.1)–(2.2) for the disciplinary analysis outputs $a_1(s, l_1, l_2)$ and $a_2(s, l_1, l_2)$. The function f represents the system-level objective.

To facilitate the introduction of CO, we have chosen a simplified model problem: each of the constraints g_i explicitly depends only a single discipline's analysis outputs. There is no constraint that involves a_1 and

a_2 jointly. Accordingly, the constraints g_1, g_2 are *disciplinary design constraints* associated with Disciplines 1 and 2, respectively. This choice of design constraints simplifies the exposition, but is by no means essential. The statement of the general problem, without this simplification, and its CO reformulations are given in the Appendix.

3. Reformulation in terms of collaborative optimization. There are several variants of collaborative optimization. We consider two specific instances in detail, with the aim of formalizing presentations of collaborative optimization that have appeared most frequently in other sources.

To reformulate (2.3) along the lines of CO, we introduce new disciplinary design variables σ_1, σ_2 that relax the coupling between the subsystems through the shared system design variables s. The variables σ_i serve as local copies (at the level of the disciplinary subproblems) of the shared variables s. In general, Greek letters will denote new, auxiliary variables designed to serve at the subproblem level as copies of shared quantities.

CO is a bilevel approach in which a system-level coordination problem attempts to optimize the system-level objective resulting in the following system-level problem:

$$\text{(3.1)} \qquad \begin{array}{ll} \underset{s,t_1,t_2}{\text{minimize}} & f(s,t_1,t_2) \\ \text{subject to} & C(s,t_1,t_2) = 0, \end{array}$$

where there are N *interdisciplinary consistency constraints* $C = \{c_1, \ldots, c_N\}$ which we describe presently. The system-level problem controls the system-level design variables s and interdisciplinary coupling variables (t_1, t_2), which, as we discuss, are system-level target values for the disciplinary inputs and outputs a_1 and a_2.

The system-level problem issues design targets (s, t_1, t_2) to the constituent disciplines. In the lower-level problems, the disciplines must design to match these targets, as follows. In Discipline 1, we are given (s, t_1, t_2) and compute $\bar{\sigma}_1(s, t_1, t_2)$ and $\bar{l}_1(s, t_1, t_2)$ as solutions of the following minimization problem in (σ_1, l_1) at the level of Discipline 1:

$$\text{(3.2)} \qquad \begin{array}{ll} \underset{\sigma_1,l_1}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_1 - s \|^2 + \| a_1(\sigma_1, l_1, t_2) - t_1 \|^2\right] \\ \text{subject to} & \quad g_1(\sigma_1, l_1, a_1(\sigma_1, l_1, t_2)) \ge 0, \end{array}$$

where a_1 is computed in this disciplinary optimization problem via the disciplinary analysis

$$a_1 = A_1(\sigma_1, l_1, t_2).$$

Note that in the disciplinary subproblem (3.2), the system-level variables (s, t_1, t_2) serve either as parameters or targets that we try to match. An analogous problem for Discipline 2 defines solutions $\bar{\sigma}_2(s, t_1, t_2)$ and $\bar{l}_2(s, t_1, t_2)$ of the problem

$$\text{(3.3)} \qquad \begin{array}{ll} \underset{\sigma_2,l_2}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_2 - s \|^2 + \| a_2(\sigma_2, l_2, t_1) - t_2 \|^2\right] \\ \text{subject to} & \quad g_2(\sigma_2, l_2, a_2(\sigma_2, l_2, t_1)) \ge 0. \end{array}$$

Again, a_2 is computed via the disciplinary analysis

$$a_2 = A_2(\sigma_2, l_2, t_1).$$

The introduction of disciplinary minimization subproblems of the form (3.2)–(3.3) is a distinctive characteristic of CO. The subproblems can be solved autonomously. By solving the subproblems, we are using the disciplinary design constraints g_i to eliminate the disciplinary design variables l_i from the system-level problem, and decoupling the calculation of the disciplinary analysis outputs a_i. Information from the solutions of the disciplinary problems (3.2)–(3.3) is then used to define the system-level consistency constraints c_i. The type of system-level constraints used gives rise to a specific variant of CO.

The first variant of CO we discuss is the one in which CO is most frequently presented. For instance, see [11, 13, 27, 28]. In this formulation, the consistency condition is to drive to zero the minimum value of the target mismatch objective in subproblems (3.2)–(3.3). At the system-level, the interdisciplinary consistency constraints are simply the optimal values of the objectives in (3.2)–(3.3). That is, the consistency constraints $C = (c_1, c_2)$ are defined as

$$c_1(s, t_1, t_2) = \frac{1}{2}\Big[\| \bar{\sigma}_1(s, t_1, t_2) - s \|^2 + \| a_1(\bar{\sigma}_1(s, t_1, t_2), \bar{l}_1(s, t_1, t_2), t_2) - t_1 \|^2\Big] \tag{3.4}$$

$$c_2(s, t_1, t_2) = \frac{1}{2}\Big[\| \bar{\sigma}_2(s, t_1, t_2) - s \|^2 + \| a_2(\bar{\sigma}_2(s, t_1, t_2), \bar{l}_2(s, t_1, t_2), t_1) - t_2 \|^2\Big], \tag{3.5}$$

where the bars over $\bar{\sigma}_1, \bar{\sigma}_2, \bar{l}_1, \bar{l}_2$ indicate that these values are the results of solving the disciplinary optimization subproblems for the given value of the system-level variables. We call this version CO_2, where the subscript "2" refers to the fact that the c_i are sums of squares.

An alternative to the system-level consistency conditions (3.4)–(3.5), giving rise to the second instance of CO we discuss, is to match the system-level variables directly with their subsystem counterparts computed in subproblems (3.2)–(3.3). The consistency constraints $C = (c_1, \ldots, c_4)$ are

$$\begin{aligned}
c_1(s, t_1, t_2) &= \bar{\sigma}_1(s, t_1, t_2) - s \\
c_2(s, t_1, t_2) &= a_1(\bar{\sigma}_1(s, t_1, t_2), \bar{l}_1(s, t_1, t_2), t_2) - t_1 \\
c_3(s, t_1, t_2) &= \bar{\sigma}_2(s, t_1, t_2) - s \\
c_4(s, t_1, t_2) &= a_2(\bar{\sigma}_2(s, t_1, t_2), \bar{l}_2(s, t_1, t_2), t_1) - t_2).
\end{aligned}$$

We denote this formulation CO_1 to indicate that the quantities in the system-level constraints are not sums of squares. Note that (c_1, c_2) are associated with Discipline 1, while (c_3, c_4) are associated with Discipline 2.

In either approach, we will call a value of the system-level variables (s, t_1, t_2) *realizable* for Discipline i if the optimal objective value in the corresponding disciplinary optimization problem (3.2) or (3.3) is zero. Realizable values of the system-level variables correspond to desirable designs: if the optimal objective value in the disciplinary optimization problem for Discipline i is zero, then this means that Discipline i can exactly match the system-level input-output targets without violating the disciplinary design constraints. In general, there will be many realizable values of the system-level variables for a given discipline. A point (s, t_1, t_2) is feasible for the system-level problem when it is realizable for all the constituent disciplines.

4. Some simple examples. To illustrate the formalism of the preceding section, we reformulate two simple optimization problems—a linear program and a convex quadratic program—along the lines of CO. The examples will be used throughout the remainder of the paper to illustrate the features of CO under consideration.

When complex problems are used as the only test of methodology, it is often difficult to distinguish the performance features due to the intrinsic properties of the method from those due to the various aspects of implementation. Using simple examples allows us to isolate the intrinsic properties of collaborative optimization. Moreover, simple problems provide a "lower bound" on reliability and robustness of a solution technique: while it is clear that MDO methods are not intended for solving very small and simple problems, any practical optimization approach should be able to solve such problems reliably.

4.1. A linear program. Consider the following optimization problem in a single variable s:

$$\begin{array}{ll} \underset{s}{\text{minimize}} & f(s) = s \\ \text{subject to} & 0 \leq s \leq 1. \end{array} \tag{4.1}$$

This, arguably, simplest optimization problem, will suffice to make a number of points about the properties of CO.

This problem has only shared design variables. To see how such a problem might come about as an MDO problem, imagine a bar of cross-sectional area A (which will be our design variable) and fixed length L, subject to a longitudinal load F. Suppose we have two contending design constraints from the "disciplines" of stress and weight. The first constraint is that the stress not exceed some allowable limit S: $F/A \leq S$. The second is that the weight not exceed some allowable limit W: $\rho LA \leq W$, where ρ is the density of the bar. As our system performance objective, we wish to minimize the total cost, which we take to be proportional to the volume of the bar: κLA, where κ is the cost per unit volume. This leads to the design problem

$$\begin{array}{ll} \underset{A}{\text{minimize}} & \kappa LA \\ \text{subject to} & F/S \leq A \\ & A \leq W/(\rho L). \end{array}$$

An appropriate change of variables leads to (4.1).

To reformulate (4.1) along the lines of CO, we create two disciplines associated with each of the two inequality constraints, as in the preceding discussion. We view the constraints $s \geq 0$ and $s \leq 1$ as two contending disciplinary design constraints. Given a value of the system-level variable s, the subsystem-level problems are then

$$\begin{array}{llcll} \underset{\sigma_1}{\text{minimize}} & \frac{1}{2} \| \sigma_1 - s \|^2 & \qquad & \underset{\sigma_2}{\text{minimize}} & \frac{1}{2} \| \sigma_2 - s \|^2 \\ \text{subject to} & \sigma_1 \geq 0 & & \text{subject to} & \sigma_2 \leq 1. \end{array} \tag{4.2}$$

The solutions of these subsystem-level problems, as functions of s, are

$$\bar{\sigma}_1(s) = \begin{cases} 0 & \text{if } s \leq 0 \\ s & \text{if } s \geq 0 \end{cases} \qquad \bar{\sigma}_2(s) = \begin{cases} s & \text{if } s \leq 1 \\ 1 & \text{if } s \geq 1. \end{cases} \tag{4.3}$$

The CO_2 reformulation is then

$$\begin{array}{ll} \underset{s}{\text{minimize}} & s \\ \text{subject to} & c_1(s) = \frac{1}{2} \| \bar{\sigma}_1(s) - s \|^2 = 0 \\ & c_2(s) = \frac{1}{2} \| \bar{\sigma}_2(s) - s \|^2 = 0, \end{array} \tag{4.4}$$

while the CO_1 reformulation is

$$\begin{array}{ll} \underset{s}{\text{minimize}} & s \\ \text{subject to} & c_1(s) = \bar{\sigma}_1(s) - s = 0 \\ & c_2(s) = \bar{\sigma}_2(s) - s = 0. \end{array} \tag{4.5}$$

4.2. A convex quadratic program. Our second example has a convex quadratic objective and linear constraints:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}(a_1^2(l_1, l_2) + 10\, a_2^2(l_1, l_2)) \\ \text{subject to} & s + l_1 \leq 1 \\ & -s + l_2 \leq -2, \end{array} \tag{4.6}$$

where $a = (a_1, a_2)$ is the solution of

$$2a_1 + a_2 = l_1$$
$$a_1 + 2a_2 = l_2.$$

Since $a = a(l_1, l_2)$ is a linear function of (l_1, l_2), the objective is a quadratic function of (l_1, l_2). Reformulating this problem along the lines of CO, we obtain the system-level problem

$$\begin{array}{ll} \underset{s,t_1,t_2}{\text{minimize}} & \frac{1}{2}(t_1^2 + 10\,t_2^2) \\ \text{subject to} & C(s,t_1,t_2) = 0. \end{array}$$

We obtain CO_2 if $C = (c_1, c_2)$ with

$$c_1(s,t_1,t_2) = \frac{1}{2}\left[\| \bar{\sigma}_1(s,t_1,t_2) - s \|^2 + \| a_1(\bar{\sigma}_1(s,t_1,t_2), \bar{l}_1(s,t_1,t_2), t_2) - t_1 \|^2\right]$$
$$c_2(s,t_1,t_2) = \frac{1}{2}\left[\| \bar{\sigma}_2(s,t_1,t_2) - s \|^2 + \| a_2(\bar{\sigma}_2(s,t_1,t_2), \bar{l}_2(s,t_1,t_2), t_1) - t_2 \|^2\right]$$

and we obtain CO_1 if $C = (c_1, \ldots, c_4)$ with

$$c_1(s,t_1,t_2) = \bar{\sigma}_1(s,t_1,t_2) - s$$
$$c_2(s,t_1,t_2) = a_1(\bar{\sigma}_1(s,t_1,t_2), \bar{l}_1(s,t_1,t_2), t_2) - t_1$$
$$c_3(s,t_1,t_2) = \bar{\sigma}_2(s,t_1,t_2) - s$$
$$c_4(s,t_1,t_2) = a_2(\bar{\sigma}_2(s,t_1,t_2), \bar{l}_2(s,t_1,t_2), t_1) - t_2,$$

where, given values of the system-level variables (s, t_1, t_2), the constrained optimal values $\bar{\sigma}_1(s,t_1,t_2)$ and $\bar{l}_1(s,t_1,t_2)$ are computed by solving the following problem for Discipline 1:

$$\begin{array}{ll} \underset{\sigma_1,l_1}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_1 - s \|^2 + \| a_1(\sigma_1, l_1, t_2) - t_1 \|^2\right] \\ \text{subject to} & \sigma_1 + l_1 \le 1, \end{array}$$

where a_1 is the solution of the disciplinary analysis

$$2a_1 + t_2 = l_1.$$

Similarly, for Discipline 2 we compute $\bar{\sigma}_2(s,t_1,t_2)$ and $\bar{l}_2(s,t_1,t_2)$, given values of the system-level variables (s, t_1, t_2), via the following problem:

$$\begin{array}{ll} \underset{\sigma_2,l_2}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_2 - s \|^2 + \| a_2(\sigma_2, l_2, t_1) - t_2 \|^2\right] \\ \text{subject to} & -\sigma_2 + l_2 \le -2, \end{array}$$

where a_2 is the solution of the disciplinary analysis

$$t_1 + 2a_2 = l_2.$$

The solutions of these disciplinary problems are

$$\bar{\sigma}_1(s,t_1,t_2) = s + 1/5\ \min((-s - 2t_1 - t_2 + 1), 0) \tag{4.7}$$
$$\bar{l}_1(s,t_1,t_2) = 2t_1 + t_2 + 4/5\ \min((-s - 2t_1 - t_2 + 1), 0) \tag{4.8}$$
$$\bar{\sigma}_2(s,t_1,t_2) = s + 1/5\ \max((-s + t_1 + 2t_2 + 2), 0) \tag{4.9}$$
$$\bar{l}_2(s,t_1,t_2) = t_1 + 2t_2 - 4/5\ \max((-s + t_1 + 2t_2 + 2), 0). \tag{4.10}$$

5. Motivation for collaborative optimization. A number of considerations motivate collaborative optimization as an alternative to the fully integrated design problem formulated in (2.3). We review some of the reasons given for CO; see [11, 12, 23, 27] for further details.

In the context of multidisciplinary design, CO is particularly motivated by considerations of both problem synthesis and problem decomposition [11, 12]. The formulation (2.3) is based on the multidisciplinary analysis (2.1)–(2.2). However, an MDA capability is not usually developed together with the constituent disciplinary analyses. CO attempts to use the latter while finessing the necessity of developing a separate MDA capability. CO is thus viewed as potentially being a facilitator of problem synthesis, with attendant savings of human effort, even if the resulting computational problem is more difficult to solve.

It is also argued that the decomposition and flow of information in CO mirror those present in engineering organizations. In the bilevel approach of CO, there is a system-level coordination problem that attempts to optimize the system-level objective. In the process of doing so, it issues design targets to the component disciplines. In the lower-level problems, the disciplines must design to match these targets. This is one sense in which CO can be viewed as respecting disciplinary autonomy.

CO is also motivated by the wish to keep the disciplinary designs feasible with respect to the corresponding disciplinary design constraints during optimization. This avoids problems with designs that cause a breakdown of disciplinary analyses. However, because the overall design (i.e., the system-level and disciplinary design variables) does not, in general, satisfy the system-level interdisciplinary consistency constraints, stopping in the middle of optimization may yield a design that is not physically consistent.

Another motivation for CO is a concern about the number of design variables in the fully integrated design problem (2.3) [27]. Fully integrated formulations lead to the presence of the complete set of local disciplinary design variables l_i in the system-level problem (2.3). The bilevel structure of CO uses the subsystem design constraints to eliminate from the system-level problem the design variables l_i local to the individual disciplines. This means that the details of the subsystem design are, in a sense, hidden in the system-level problem. Note that this elimination is done via the disciplinary optimization problems.

Finally, CO is based on the intuition that there should be little interaction among the local design variables of different disciplines, which suggests that the system-level coordination problem could be solved quickly and easily.

6. Delinquent features of CO. In this section we discuss and illustrate by simple examples the most pronounced analytical features of the system-level and disciplinary optimization problems and their computational consequences. We believe that the analysis explains many of the computational difficulties reported in [2, 22, 36]. In the light of these analytical characteristics, we later re-examine the rationale for collaborative optimization.

Our analysis focuses on difficulties that *necessarily* arise at points that are realizable for individual disciplines, and, more specifically, designs that satisfy the system-level consistency constraints, i.e., designs and disciplinary inputs and outputs that correspond to a consistent multidisciplinary analysis. These difficulties include:

1. The standard Karush-Kuhn-Tucker conditions for a solution do not hold for the CO_2 system-level problem. That is, Lagrange multipliers do not exist for the system-level equality constrained problem that results in CO_2.
2. The derivatives of the CO_1 system-level constraints will be discontinuous at values of the system-level variables that are realizable for a given discipline if the solution of the disciplinary optimization problem is on the boundary of the disciplinary feasible region. Unfortunately, this means that, in

general, the Jacobian of the system-level constraints is discontinuous at solutions of the system-level optimization problem, and the standard Karush-Kuhn-Tucker conditions for a solution do not hold for the CO_1 system-level problem.

3. The system-level optimization problems in CO are more nonlinear than the fully integrated formulation of §2.3. For instance, collaborative optimization transforms linear programs into nonlinear programs.

All of these features make it harder for conventional optimization algorithms to solve the optimization problems in CO, and will also degrade the efficiency with which these problems will be solved, if they can be solved at all. That is, the nature of the system-level consistency constraints makes it very difficult or impossible for a conventional optimization algorithm to find a solution that satisfies the design constraints of the original problem. Moreover, bad things happen at good points: as we discuss, the difficulties arise at values of the system-level variables that are realizable. These computational difficulties are an unavoidable consequence of the bilevel approach that CO uses to eliminate the disciplinary design variables from the system-level problem and manifest disciplinary autonomy.

The proofs of the mathematical propositions are technical and not germane to the discussion in this paper. Instead, we illustrate the effects via examples. Interested readers may find the proofs in [3].

6.1. Breakdown of the standard stationarity conditions in CO_2. The system-level problem in CO_2 fails to satisfy the standard Karush-Kuhn-Tucker stationarity conditions for a constrained minimizer. As we discuss, with rare exceptions Lagrange multipliers do not exist for the equality constrained system-level problem that arises in CO_2.

The simple example of §4.1 suffices to illustrate the breakdown of the stationarity conditions. From (4.3) and (4.4) we see that the gradients of the system-level consistency constraints are given by

$$\nabla c_1(s) = \begin{cases} s & \text{if } s \le 0 \\ 0 & \text{if } s \ge 0 \end{cases} \qquad \nabla c_2(s) = \begin{cases} 0 & \text{if } s \le 1 \\ s & \text{if } s \ge 1. \end{cases}$$

The minimizer of the system-level problem (4.4) is $s_* = 0$, and

$$\nabla c_1(s_*) = \nabla c_2(s_*) = 0.$$

These constraint gradients are linearly dependent and hence violate the standard constraint qualification (see, e.g., [14,17] for detailed discussions of constraint qualifications). The normal stationarity conditions for (4.4) would require the existence of Lagrange multipliers λ_1, λ_2 such that

$$\nabla f(s_*) + \lambda_1 \nabla c_1(s_*) + \lambda_2 \nabla c_2(s_*) = 0.$$

However, we have

$$\nabla f(s_*) + \lambda_1 \nabla c_1(s_*) + \lambda_2 \nabla c_2(s_*) = \nabla f(s_*) = 1. \tag{6.1}$$

Thus the normal stationarity conditions cannot be satisfied at s_*.

The breakdown of the standard Karush-Kuhn-Tucker conditions in the system-level problem that arises in CO_2 is a general feature of the CO_2 approach. It is a consequence of the system-level constraint Jacobian vanishing at all points that are feasible with respect to the system-level equality constraints. To understand this, consider a generic equality constrained minimization problem:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & C(x) = 0, \end{array} \tag{6.2}$$

where $C(x) = (c_1(x), \ldots, c_m(x))$. The associated Karush-Kuhn-Tucker necessary condition for a point x_* to be a (local) minimizer of (6.2) is that $C(x_*) = 0$ and that there exist Lagrange multipliers $\lambda_* = (\lambda_1, \ldots, \lambda_m)$ for which

$$\nabla f(x_*) + \nabla C(x_*)\lambda_* = 0, \tag{6.3}$$

where $\nabla C(x)$ is the transpose of the Jacobian of C. However, as the following proposition illustrates, it is entirely possible that the Karush-Kuhn-Tucker conditions do not hold at a solution.

Proposition 6.1. *Let x_* be feasible for (6.2), and suppose that $\nabla C(x_*) = 0$. Then the stationarity condition (6.3) holds if and only if $\nabla f(x_*) = 0$.*

This follows immediately from (6.3): $\nabla f(x_*) + \nabla C(x_*)\lambda_* = \nabla f(x_*)$, since $\nabla C(x_*) = 0$. Thus, if the constraint Jacobian vanishes at x_*, the Lagrange multiplier rule will not hold at x_*, unless x_* is also an unconstrained stationary point: $\nabla f(x_*) = 0$. In general, this is not the case.

Unfortunately, the situation described by Proposition 6.1 necessarily arises in the system-level problem of CO_2. The system-level constraints are differentiable at system-level feasible points; however, the Jacobian is zero. More precisely, at values of the system-level variables that are realizable for a given discipline, the gradient of the CO_2 system-level constraints associated with that discipline vanishes.

Proposition 6.2. *Let $c_i(s, t_1, t_2)$ be the CO_2 system-level constraint associated with Discipline i. Then, if $c_i(s, t_1, t_2) = 0$, we have $\nabla c_i(s, t_1, t_2) = 0$.*

Proposition 6.2 means that the Jacobian of the system-level constraints will drop rank whenever the system-level variables become realizable for one or more of the disciplines. This can cause numerical optimization applied to the system-level problem to fail at realizable values of the system-level variables. Another consequence is the following.

Corollary 6.3. *The Jacobian of the system-level equality constraints in CO_2 vanishes at every feasible point of the system-level problem.*

In view of this corollary, Proposition 6.1 says that Lagrange multipliers *do not exist,* in general, for the system-level problem in CO_2. While solutions to the system-level problem will exist, we cannot identify them, and this inability to characterize solutions numerically via the Karush-Kuhn-Tucker conditions has unfortunate consequences for computation. Assumptions about the validity of the stationarity conditions underlie the ways in which optimization algorithms compute steps, gauge progress, and make decisions about termination, among other things.

The nonexistence of Lagrange multipliers manifests itself in a number of practical difficulties. One curious feature is that one could begin an optimization algorithm at or near the solution to the original system-level problem, but, because the usual stationarity conditions for an optimizer do not hold, the algorithm will move away from the solution, leaving the feasible region, and return to it only later.

A related feature is that algorithms that use augmented Lagrangians or similar merit functions to decide whether to accept an iterate can break down due to the fact that certain parameters, such as the penalty weights in the problem merit function, are growing without bound. The divergence of these parameters is a consequence of the fact that near the feasible region, the constraints are flat to first order while the objective is not, in general.

Moreover, since the usual stationarity condition does not hold at solutions of the system-level optimization problem, we have no way to gauge the progress of a conventional optimization algorithm applied to the system-level problem. Once the optimization algorithm terminates, we cannot, say, look at the size of the gradient of the Lagrangian to determine whether we are close to a minimizer.

Although the nonexistence of Lagrange multipliers is an inherent feature of CO_2, this difficulty is not due to the *intrinsic geometry* of the system-level or disciplinary feasible regions. Rather, the problem lies in the way the feasible region is *represented* in terms of system-level constraints in CO_2—the CO_2 formulation introduces this analytical problem. The vanishing of the Jacobian has been observed previously [11], but its full consequences appear not to have been fully appreciated.

Simple numerical tests illustrate how this analytical feature of CO_2 can impede and even thwart computational optimization. Table 6.1 presents the behavior of a sequential quadratic programming algorithm applied to the one-variable problem

$$\begin{array}{ll} \underset{s}{\text{minimize}} & s \\ \text{subject to} & 0 \leq s \leq 1, \end{array}$$

reformulated along the lines of CO_2 as discussed in §4.1. These results were obtained using the NPSOL[1] optimization package [20] with analytic first-derivatives of the objective and constraints. The solution of the original problem is $s_* = 0$. The initial guesses for the two tests are $s_0 = 0.001$, which is close to the solution s_* and is also feasible with respect to the system-level constraints, and $s_0 = -0.001$, which is also near the solution but slightly infeasible.

Iteration	s	Penalty	Cumulative work
0	1.000e-03	0.0e+00	1
1	-9.990e-01	4.2e+00	2
2	-9.847e-01	5.7e+00	4
3	-8.282e-01	7.4e+00	6
4	-4.142e-01	2.7e+01	7
5	-3.430e-01	5.9e+01	9
6	-1.718e-01	4.0e+02	10
7	-1.436e-01	8.2e+02	12
8	-7.251e-02	5.4e+03	13
9	-6.076e-02	1.1e+04	15
10	-3.203e-02	6.5e+04	16
11	-2.717e-02	1.2e+05	18
12	-1.727e-02	5.1e+05	19
13	-1.442e-02	1.9e+06	20
14	-1.414e-02	4.7e+06	21

Iteration	s	Penalty	Cumulative work
0	-1.000e-03	0.0e+00	1
1	-1.000e-00	1.0e+00	2
2	-9.857e-01	1.4e+00	4
3	-8.290e-01	1.9e+00	6
4	-4.145e-01	6.9e+00	7
5	-3.432e-01	1.5E+01	9
6	-1.716e-01	1.0E+02	10
7	-1.434e-01	2.1E+02	12
8	-7.170e-02	1.4E+03	13
9	-5.992e-02	2.8E+03	15
10	-2.996e-02	1.9E+04	16
11	-2.503e-02	3.9E+04	18
12	-1.252e-02	2.6E+05	19
13	-1.046e-02	5.3E+05	21
14	-5.230e-03	3.5E+06	22

FIG. 6.1. *Iteration history of* NPSOL *applied to the CO_2 system-level problem for a one-variable linear program with two starting points,* $s_0 = 0.001$ *and* $s_0 = -0.001$.

When we start at $s_0 = 0.001$, because the system-level equality constraints vanish in the interior of the feasible region $0 \leq s \leq 1$, the problem appears to be unconstrained at s_0 and so we immediately take a large step that produces s_1 violating the design constraints. We then spend the remainder of the iterations working our way back towards feasibility.

Note also the column labeled "Penalty". This is the value of the penalty weight in the augmented Lagrangian used by NPSOL as a merit function by which to gauge progress. The fact that the penalty

[1] The use of names of commercial software in this paper is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products by the National Aeronautics and Space Administration or ICASE.

weight increases to such a large value reflects the fact that Lagrange multipliers do not exist for the system-level CO_2 problem; in effect, the algorithm is compensating for the system-level constraint Jacobian vanishing by increasing the penalty parameter (in principle, without bound as it approaches the solution).

Finally, note the increased non-linearity introduced by the CO_2 reformulation. The original problem is a trivial linear program that NPSOL solves in a single iteration. However, the CO_2 reformulation requires many more iterations and much more work, as reflected in the column "Cumulative work," which is a running tally of the number of evaluations of the system-level objective function. Each evaluation requires the solution of the disciplinary optimization problems.

The behavior of NPSOL applied to the system-level problem that results from the CO_2 formulation of the convex quadratic program (4.6) is even more striking[2]. The solution to the original problem is unique, since the original problem is strictly convex. From some starting points, (e.g., $(s, t_1, t_2) = (1, 1, 1)$) NPSOL applied to the CO_2 system-level problem finds the optimal solution:

$$\begin{gathered} s = 18/11 = 1.63\overline{63}, \qquad t_1 = -10/33 = -0.30\overline{30}, \quad t_2 = -1/33 = -0.03\overline{03}, \\ \bar{\sigma}_1(s, t_1, t_2) = \bar{\sigma}_2(s, t_1, t_2) = s, \qquad \bar{l}_1(s, t_1, t_2) = -7/11, \qquad \bar{l}_2(s, t_1, t_2) = -4/11, \\ a_1(s, t_1, t_2) = t_1, \quad a_2(s, t_1, t_2) = t_2, \end{gathered} \tag{6.4}$$

with the associated optimal value function of 5.05×10^{-2}, although at a considerably greater computational cost (several hundred disciplinary analyses) than that of the solution of the fully integrated formulation. None of the iterates generated in the system-level problem are realizable for either discipline. In light of Proposition 6.2, this non-realizability is actually the favorable situation from the perspective of applying a numerical optimization algorithm to the CO system-level problem. However, this is at odds with what would make most sense from an engineering perspective, since intermediate designs do not satisfy the physical constraints.

On the other hand, the worst sort of behavior is observed starting from values of the system-level variables that are feasible with respect to the system-level consistency constraints (which is at odds with what one would hope for). For instance, if we start at $(s, t_1, t_2) = (-3, -3, -3)$, NPSOL terminates after a few system-level iterations (and a cost of over 200 analyses for each discipline) at

$$s = -2.806, \quad t_1 = -5.658, \quad t_2 = 0.301 \tag{6.5}$$

with the associated objective value of 16.46. This final value of the system-level variables (and all intermediate iterates) satisfy the system-level equality constraints. Unfortunately, this is the adverse situation in collaborative optimization, since the computed Jacobians at these point are singular, and this causes NPSOL to fail. This singularity is reflected in the final estimate of -1.67×10^{10} for the Lagrange multiplier associated with the system-level constraint c_1.

Even if we begin much closer to the solution, but still feasible with respect to the system-level constraints, we may fail to converge to the solution. For instance, starting from $(s, t_1, t_2) = (1.63, -0.3333, -0.0333)$, NPSOL applied to the system-level problem halts, unable to make further progress, at a feasible value of the system-level variables with an objective of 6.10×10^{-2}, over 20% greater than the optimal value, and only a slight improvement on an initial objective value of 6.11×10^{-2}.

As noted previously, because the usual constrained stationarity conditions do not hold at a solution of the system-level problem in CO_2, we cannot reliably use the conventional metrics to determine whether

[2] In the CO_2 tests described, we compute system-level sensitivities via post-optimality sensitivity analysis of solutions of the disciplinary problems, as one might in practical application of CO_2, while the sensitivities used inside the disciplinary subproblems are computed analytically.

these spurious answers are nearly stationary and thus close to a solution. For instance, if we start with the system-level values $(s, t_1, t_2) = (1.63, -0.302, -0.302)$, we terminate at a point that appears to NPSOL to be a Karush-Kuhn-Tucker point. That is, the point is feasible and the projection of the objective gradient onto the linearization of the active constraints is small in magnitude. However, the Jacobian is nearly zero, so small errors in computing the system-level sensitivities makes the Jacobian mostly noise. Thus, the projection is bogus. While the original problem has a unique minimizer, we cannot not find it reliably in the CO_2 reformulation.

We stress that this is not the fault of NPSOL. Instead, it is a feature of the CO_2 approach, which formulates the system-level problem so that the Jacobian of the CO_2 system-level constraints necessarily vanishes at all feasible points (and will be singular at realizable points). For instance, in [36], the authors applied both a feasible direction and a penalty function algorithm to a version of CO_2. They also found that the designs at which their algorithms terminated were highly variable and very sensitive to the choice of parameters in the optimization algorithms.

These examples show that CO_2 takes very simple, smooth, convex optimization problems with small numbers of variables and transforms them into problems that are difficult to solve. Moreover, the analytical reasons for the computational difficulties are inherent to the approach, so they will not disappear for problems of greater size or complexity.

6.2. Breakdown of the standard stationarity conditions in $\mathbf{CO_1}$. The CO_1 version of CO is motivated by the need to alleviate the performance problems of CO_2 [11]. However, the use of CO_1 presents its own difficulties. In this section, we discuss how it is almost always the case that the Jacobian of the CO_1 system-level constraints will be discontinuous at a solution. More precisely, the derivatives of the CO_1 system-level consistency constraints associated with a given discipline are *necessarily* discontinuous at the boundary of the feasible region for that discipline.

The examples in §4 manifest this problem. For instance, for the linear program (4.1), from (4.3), we see that the system-level constraints are

$$c_1(s) = \bar{\sigma}_1(s) - s = \begin{cases} -s & \text{if } s \leq 0 \\ \ \ 0 & \text{if } s \geq 0 \end{cases}, \qquad c_2(s) = \bar{\sigma}_2(s) - s = \begin{cases} 0 & \text{if } s \leq 1 \\ 1 - s & \text{if } s \geq 1 \end{cases},$$

both of which have discontinuous derivatives, the first at $s = 0$ and the second at $s = 1$. These points correspond to the boundaries of the disciplinary feasible regions $\{ s \mid s \geq 0 \}$ and $\{ s \mid s \leq 1 \}$. A similar lack of differentiability can be seen in the CO reformulation of the quadratic program in §4.2. The solutions of the disciplinary subproblems are given in (4.7)–(4.10); the presence of the min and max terms makes these solutions non-differentiable at values of the system-level variables along the boundary of the realizable sets for each discipline.

This discontinuity of the system-level derivatives is a general feature of CO_1 and is not peculiar to the examples in §4. Each subsystem-level problem (3.2)–(3.3) minimizes the distance from the disciplinary feasible region to the target values of the system-level design and coupling variables. For target values of the system-level variables corresponding to designs at the boundary of a disciplinary feasible region, the solution of the corresponding disciplinary optimization problem undergoes an abrupt and non-differentiable change. The following proposition gives a mathematical statement of this observation.

Proposition 6.4. *Let $C_i(s, t_1, t_2)$ be the set of CO_1 system-level consistency constraints associated with Discipline i. Suppose (s, t_1, t_2) is realizable for Discipline i, and C_i is defined on a neighborhood of (s, t_1, t_2). If the solution $\bar{\sigma}_i(s, t_1, t_2)$, $\bar{l}_i(s, t_1, t_2)$ is on the boundary of the disciplinary feasible region (i.e., one or more* *of the disciplinary design constraints are binding), then the Jacobian of C_i is discontinuous at (s, t_1, t_2).*

Moreover, this discontinuity of the derivatives will, in general, occur *at the solution to the system-level problem*. This is so because, in general, at least one disciplinary design constraint will be binding at the solution. This means that we can expect the solution of the system-level problem to be on the boundary of one (or more) of the feasible regions for the individual disciplines, and at such points the CO_1 constraints have discontinuous derivatives.

The convex quadratic program in §4.2 demonstrates the effects of the discontinuity of the system-level consistency constraint Jacobian. NPSOL finds the solution of the original problem in 4–5 iterations, regardless of the starting point. (For the purposes of comparison we treated the linear constraints as general nonlinear constraints.)

NPSOL applied to the CO_1 reformulation, on the other hand, behaves erratically. If we start from the initial point $(s, t_1, t_2) = (1.63, -0.302, -0.0302)$, which is close to the exact solution, $(1.\overline{63}, -0.\overline{30}, -0.\overline{03})$, NPSOL finds the solution at a cost of about 50 disciplinary optimization problems for each discipline.

On the other hand, if we start at the point $(s, t_1, t_2) = (-1, -1, -1)$, NPSOL terminates, unable to make further progress, at

$$
\begin{gathered}
s = -0.996872, \qquad t_1 = -1.463996, \qquad t_2 = -0.06887874 \\
\bar{\sigma}_1(s, t_1, t_2) = \bar{\sigma}_2(s, t_1, t_2) = s, \quad \bar{l}_1(s, t_1, t_2) = -1.601754, \quad \bar{l}_2(s, t_1, t_2) = -2.996872, \\
a_1(s, t_1, t_2) = t_1, \quad a_2(s, t_1, t_2) = t_2.
\end{gathered}
$$

This design satisfies the system-level constraints but has an associated objective value of 1.096 (the optimal objective value is 5.05×10^{-2}). The associated disciplinary design variables for Discipline 2 also lie on the boundary of the disciplinary feasible region (since $s + \bar{l}_2 = -2$), so we encounter the discontinuity in the constraint Jacobian described in Proposition 6.4. Examination of the finite-difference estimate of the Jacobian computed by NPSOL at this point reveals that the Jacobian is highly inaccurate.

Starting from $(s, t_1, t_2) = (0, 0, 0)$, NPSOL approaches but does not succeed in finding the correct answer of the original problem. We terminated this run after 500 system-level iterations with an objective that was 2.7 times greater than the optimal value, at a cost of solving over 3000 disciplinary optimization problems for each discipline.

Again, these difficulties are not the fault of NPSOL. Collaborative optimization produces system-level problems that defeat traditional smooth optimization algorithms. In this case, the Jacobian of the CO_1 system-level constraints is discontinuous at realizable designs on the boundary of one or more disciplinary feasible regions.

6.3. Additional sources of nonsmoothness. There are additional sources of nonsmoothness that can arise in collaborative optimization. In particular, when the set of disciplinary design constraints that are binding at the solution of the disciplinary optimization problems (3.2)–(3.3) changes as a function of the system-level targets (s, t_1, t_2), there may be a discontinuity in the derivative of the system-level constraints for both CO_1 and CO_2. In connection with bilevel approaches, this difficulty was previously noted in, e.g., [8–11], and is a well-known phenomenon that arises in studying the dependence of solutions of optimization problems on parameters. The discontinuity of the constraint Jacobian discussed in §6.2 is one manifestation of this phenomenon that necessarily arises in CO.

There is also the potential for multiple local solutions of the disciplinary subproblems (3.2) and (3.3). The computed solutions may fail to depend continuously on the system-level targets (s, t_1, t_2). It is not clear how to insure that one computes only a continuous branch of local minimizers to the subsystem problems.

We do not illustrate these two difficulties with examples. They are widely known, and they may or may not occur, depending on the particular problem and the algorithms applied to its solution. The characteristics described in the previous sections, on the other hand, are intrinsic to CO and may occur even if the original problem is perfectly well-behaved.

6.4. Overdetermined system-level constraints. In a typical, well-posed equality constrained minimization problem, there are fewer equality constraints than optimization variables, i.e., the degrees of freedom in searching for a problem solution outnumber the binding constraints. Both CO_1 and CO_2 can result in system-level optimization problems that have more equality constraints than optimization variables, with CO_1 typically leading to more system-level equality constraints than does CO_2. The simple one-variable linear program illustrates this: the system-level problem has a single variable but two equality constraints.

While the system-level equality constraints are consistent insofar as they are satisfied at any multidisciplinary consistent design (s, t_1, t_2), away from the solution the overdetermined system-level constraints may cause trouble for standard optimization algorithms. For instance, in an sequential quadratic programming (SQP) approach, the system-level constraints are linearized. Since there may be many more constraints than unknowns, the resulting linear system may appear to be overdetermined and without solution, leading to an infeasible SQP subproblem. This, in turn, may lead the optimization algorithm to wrongly conclude that the system-level problem is infeasible and to terminate without having found a solution. We have observed this behavior in practice (although, to its credit, NPSOL generally does not fail because of the overdetermined constraints.)

In CO_2, one can try to avoid this problem by summing the constraints from different disciplines into a single system-level nonlinear equality constraint. In CO_1, this remedy does not apply.

6.5. System-level sensitivities. In collaborative optimization, the sensitivities of the system-level consistency constraints involve sensitivities of the solutions of the subsystem-level optimization problems (3.2) and (3.3). Although, in general, collaborative optimization is intended for problems in which the interdisciplinary coupling variables are significantly fewer in number than the local disciplinary variables, so that the CO system-level problem has fewer variables than does a fully integrated formulation, the cost of computing sensitivities for the system-level problem is still an issue due to the cost of the disciplinary optimization problems that define the system-level constraints.

In general, at non-realizable points one can compute the CO_2 system-level constraint sensitivities relatively inexpensively via standard post-optimality sensitivity analysis of solutions of the disciplinary optimization problems (e.g., [15,17]). On the other hand, analytical computation of the sensitivities of the system-level problem in CO_1 requires the second derivatives with respect to all the disciplinary design variables (σ_i, l_i) of the objective and constraints in the disciplinary problems (3.2)–(3.3) (see, e.g., [15,17]). This makes analytical computation of the requisite system-level sensitivities impractical. Instead, one would need to rely on finite-difference estimates of the system-level sensitivities. This, in turn, potentially limits the applicability of CO_1 to problems whose interdisciplinary coupling has a very limited bandwidth, as measured in the number of system-level design variables s and system-level input-output variables (t_1, t_2).

6.6. Increased nonlinearity of the transformed problem. The simple examples of §4 show that the system-level problem that one obtains in collaborative optimization will be more complicated and nonlinear than the original problem. For example, the original problem (4.1) is linear—a linear objective with linear constraints. However, the resulting CO subsystem problems in (4.2) are nonlinear. In the case of CO_2, the resulting system-level problem involves equality constraints that are piecewise quadratic, while in the case of CO_1, the system-level problem has equality constraints that are not continuously differentiable.

The increased nonlinearity arises from the elimination of the local, disciplinary design variables via the disciplinary optimization subproblems. CO transforms originally smooth problems into nonsmooth ones with a higher degree of nonlinearity. As a general rule in nonlinear programming, it is important not to increase nonlinearity or introduce other structural complications when transforming problems [19, 21]. For instance, practical optimization algorithms are often based on successive linearizations. Linearization is exact for linear problems, which allows algorithms to take advantage of this special structure. Collaborative optimization reformulation does not preserve linearity in the system-level constraints.

6.7. Infeasibility of the disciplinary problems. The presentation of collaborative optimization in §3 formalizes presentations that have appeared elsewhere (e.g., [11]). However, this problem statement does not guarantee the feasibility of the disciplinary subproblems (3.2) and (3.3). For instance, the disciplinary problem for Discipline 1,

$$\begin{array}{ll} \underset{\sigma_1, l_1}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_1 - s \|^2 + \| a_1(\sigma_1, l_1, t_2) - t_1 \|^2\right] \\ \text{subject to} & g_1(\sigma_1, l_1, a_1(\sigma_1, l_1, t_2)) \geq 0, \end{array}$$

may fail to be feasible for certain values of the system-level coupling variables t_2. That is, given t_2, there may be no values of the local variables (σ_1, l_1) for which $g_1(\sigma_1, l_1, a_1(\sigma_1, l_1, t_2)) \geq 0$. The possibility of subproblem in feasibility was noted in [7] but no alternatives were proposed.

This problem can be addressed by introducing local copies τ_j of the coupling variables t_j. We are led to disciplinary subproblems of the following form. For Discipline 1, we have

$$\begin{array}{ll} \underset{\sigma_1, l_1, \tau_2}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_1 - s \|^2 + \| \tau_2 - t_2 \|^2 + \| a_1(\sigma_1, l_1, \tau_2)) - t_1 \|^2\right] \\ \text{subject to} & g_1(\sigma_1, l_1, a_1(\sigma_1, l_1, \tau_2)) \geq 0, \end{array}$$

where a_1 is computed in this disciplinary optimization problem via the disciplinary analysis

$$a_1 = A_1(\sigma_1, l_1, \tau_2).$$

The variable τ_2 is a disciplinary stand-in for a_2. An analogous problem for Discipline 2 defines solutions $\bar{\sigma}_2(s, t_1, t_2)$, $\bar{l}_2(s, t_1, t_2)$, and $\bar{\tau}_1(s, t_1, t_2)$ of the problem

$$\begin{array}{ll} \underset{\sigma_2, l_2, \tau_1}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_2 - s \|^2 + \| \tau_1 - t_1 \|^2 + \| a_2(\sigma_2, l_2, \tau_1) - t_2 \|^2\right] \\ \text{subject to} & g_2(\sigma_2, l_2, a_2(\sigma_2, l_2, \tau_1)) \geq 0. \end{array}$$

Again, a_2 is computed via the disciplinary analysis

$$a_2 = A_2(\sigma_2, l_2, \tau_1).$$

Here the variable τ_1 stands in for a_1.

Another advantage of this formulation is that it simplifies the calculation of the sensitivities of the system-level constraints via post-optimality sensitivity analysis of the disciplinary subproblems. However, this version of collaborative optimization still suffers from the analytical problems discussed previously.

7. Relaxation of system-level constraints. As we have seen, the bilevel nature of collaborative optimization makes it difficult for numerical optimization algorithms to arrive at realizable, interdisciplinary consistent designs. In order for a practical optimization algorithm to solve an MDO problem formulated in terms of CO, one must be careful to stay away from physically consistent designs (i.e., realizable values of the system-level variables).

One relaxation is to treat the system-level interdisciplinary consistency constraints as inequalities rather than strict equalities (e.g., [22]). Tolerances on those inequalities should be as loose as possible to prevent breakdown of numerical optimization algorithms. At the same time, one must continue to impose as tight a tolerance as possible on the convergence of the disciplinary optimization subproblems, so that the system-level constraints and their sensitivities are properly evaluated.

This approach has flaws. If the system-level equality constraints are not satisfied, the system-level variables correspond to a design that is not physically consistent (because the multidisciplinary analysis relations are not satisfied) and that also violates the design constraints of one or more disciplines.

Furthermore, this relaxation is not guaranteed to relieve the computational problems, or to lead to designs that are "nearly" optimal. This is illustrated by the CO_2 formulation of the convex quadratic program (4.6). We relax the system-level constraints to be

$$\begin{aligned} c_1(s, t_1, t_2) &\leq \varepsilon \\ c_2(s, t_1, t_2) &\leq \varepsilon \end{aligned}$$

for $\varepsilon > 0$. From (4.7)–(4.10), one can show that the relaxed system-level problem is a strictly convex program, which accordingly has a unique minimizer.

First suppose we begin at $(s, t_1, t_2) = (-3, -3, -3)$, which led to the false solution (6.5). The results for different values of ε are reported in Fig. 7.1. In all cases NPSOL fails to finds a legitimate solution for the relaxed system-level problem, even though the relaxed program is convex. Again, the algorithm is undone by the singularity of the constraint Jacobian at feasible and realizable values of the system-level variables (i.e., rows of the Jacobian vanish and/or become linearly dependent).

ε	Final objective	KKT point for relaxed problem?
10^{-1}	4.75	No
10^{-2}	4.79	No
10^{-3}	4.79	No
10^{-4}	4.79	No
10^{-5}	4.79	No
10^{-6}	4.79	No

FIG. 7.1. *Results of relaxing the CO_2 constraints for the convex quadratic program example, starting at* $(-2, -2, -2)$.

On the other hand, if we repeat the same experiment starting at $(s, t_1, t_2) = (1, 1, 1)$, which allowed us to find the correct solution (6.4), we encounter another pitfall. The results are summarized in Fig. 7.2. This time, the relaxed constraints allow NPSOL to find objective values that are significantly better than the true value 5.05×10^{-2}.

Whether such deviations are acceptable in a realistic problem depends on the application. If the objective function is a physical value, such as range of an aircraft, the deviation may be significant. By relaxing the system-level consistency tolerances, we are now sometimes able to solve the wrong problem more easily.

As this simple example makes clear, relaxing the system-level constraints does not necessarily fix CO_2, and may introduce another difficulty. It is not clear that there is an ideal relaxed tolerance for the system-level constraints that allows one to solve the problem while not distorting the solution to unacceptable levels.

ε	Final objective	KKT point for relaxed problem?
10^{-1}	1.62×10^{-20}	Yes
10^{-2}	6.82×10^{-3}	Yes
10^{-3}	3.23×10^{-2}	Yes
10^{-4}	4.43×10^{-2}	Yes
10^{-5}	4.85×10^{-2}	Yes
10^{-6}	4.99×10^{-2}	Yes

FIG. 7.2. *Results of relaxing the CO_2 constraints for the convex quadratic program example, starting at* $(1,1,1)$.

Furthermore, one would expect this sensitivity to relaxation to be greater in more complex problems. In general, one cannot know, *a priori*, the effect of such relaxations on the optimal solution, since knowledge would require a perturbation analysis at the optimal solution.

In our simple tests, we varied the tolerances parametrically. In practical engineering problems, users may have a physics-based or an engineering-based idea of acceptable variations in the design variables, objectives, or constraints. For example, variations of one pound may be acceptable in the gross lift-off weight for an aircraft. However, in order to translate an acceptable variation in a physical quantity to an acceptable tolerance for infeasibility of the system-level constraints, one would need an estimate of the associated Lagrange multipliers at an optimal solution, which is not available *a priori*. Of course, we knew exact solutions to our simple example problems and were thus able to judge the distance of any iterate from the solution. Estimates of distance from solutions are unavailable in practice.

Response surface methodology (RSM) has been proposed as another approach to relaxing CO [24, 29, 30]. In one technique, the disciplinary analyses serve to build response surfaces that replace the analyses as function evaluators in the subproblems. This is a conventional use of RSM and it does not alleviate the analytical difficulties of CO: the optimization problem structure remains unchanged (although the additional uncertainty of the quality of RSM approximation now contributes to the formulation).

The second approach uses RSM to build response surfaces that attempt to approximate the constrained optimal value functions of the disciplinary subproblems. A detailed analysis of this approach would depend both on the problem and on the specific type of data fitting surfaces used. As we are dealing with the fundamental CO formulation in this paper, such an analysis is not in its scope. However, the conceptual difficulty of the approach can be summarized as follows. Although response surfaces do smooth out the problem and ease the solution by virtue of distancing the problem from the original CO formulation, it then becomes difficult to say what problem we are actually solving. On the other hand, the better the response surfaces approximate the problem, the more difficult it becomes to solve it, as the numerical properties of the approximation begin to approach those of the real problem.

Approximation techniques are usually applied to well-posed problems to help reduce the computational expense. Although computational cost is a concern in CO, the use of response surfaces in connection with collaborative optimization is mainly aimed at avoiding difficulties with CO, not difficulties with the MDO problem itself. Thus, it may not make sense to employ approximation techniques to alleviate intrinsic structural flaws of a formulation.

8. Lessons learned.

We have shown that CO formulations give rise to nonlinear programming problems that are difficult or impossible to solve by conventional optimization algorithms. The difficulty is due not to intrinsic properties of the original MDO problem, but rather to the bilevel representation in CO.

As simple examples show, if a CO formulation can be solved at all, it is unlikely that it will be solved quickly. Thus, computational efficiency is not one of the method's attributes, in general. Depending on the application, the lack of efficiency may be a deciding factor in selecting a problem formulation. Requirements for inclusion of expensive, high-fidelity analyses may preclude the use of CO in a practical environment [18]. Even if computational efficiency were not an important consideration, one should still examine other characteristics when deciding on a formulation. With the analysis of the preceding sections in mind, we comment on some of the remaining rationale for CO and on additional considerations that should assist a user in choosing a problem formulation appropriate for a particular application.

8.1. Robustness. The concept of robustness in nonlinear programming has two major components. First, it denotes the ability of an algorithm to find an answer, starting from an arbitrary initial point. Second, a robust algorithm assures that the answer found is correct and, if not, terminates with an informative message. The second feature is, arguably, by far the more important of the two. Since algorithms operate on the optimization problems that ultimately result from formulations of MDO problems, robustness is closely related to non-degeneracy of a formulation [16]. CO formulations are not well-posed from the perspective of conventional nonlinear programming because the system-level problems do not satisfy the standard optimality conditions.

An algorithm applied to the system-level problem in collaborative optimization cannot be expected to exhibit robust behavior. An algorithm's inability to find solutions is less serious than its inability to recognize one. Our analytical results and computational examples show that CO, when started at or near solutions, will generally leave the region. The lack of a verifiable stopping criterion for CO also explains the tendency of NLP software to halt at points that are not solutions, because progress cannot be made. To provide a reliable stopping criterion in terms of the standard problem formulation (2.3) one would have to implement the standard formulation, which would defeat the purpose of implementing collaborative optimization in the first place. This same reasoning also prevents uses of a hybrid approach, e.g., collaborative optimization combined with the fully integrated formulation (2.3).

We have also shown that attempts to relax the CO formulation may not alleviate the computational difficulties. Relaxation may not make it possible to find the solution reliably, and can also lead to answers that appear significantly better than the correct solution but violate the physical consistency of the multidisciplinary analysis and violate disciplinary constraints.

Design improvement, not optimality, may guide a user in the course of optimization, especially because problem expense and complexity frequently dictate that only a limited number of iterations can be performed. This approach leads to its own difficulties, because in constrained optimization progress is measured algorithmically via merit functions that balance optimality and feasibility. Thus, at any intermediate iteration of solving the system-level problem, unless a conventional convergence criterion is met, improvement in the merit function does not guarantee an overall improvement in the design objective or a nearly optimal design.

These features add up to what we consider to be the most serious drawback of CO—the lack of robustness in using nonlinear programming algorithms to solve the CO formulations. Again, this is due to the nature of the system-level problem in CO, not to the nature of the original physical problem, nor to deficiencies of standard optimization algorithms.

8.2. Disciplinary autonomy. The autonomy of disciplinary optimization is one of the strongest motivational factors for CO. Indeed, collaborative optimization achieves a marked degree of disciplinary autonomy, and the elimination of the local disciplinary design variables from the system-level optimization problem is an attractive feature of CO.

The disciplinary subproblems are not performing disciplinary optimization in the single-discipline sense. That is, the subproblem objective functions are not disciplinary objectives, such as lift, drag, or weight. Instead, the disciplinary objectives serve to minimize the inconsistency among the disciplinary analyses or, in other words, they arrive at the MDA in a distributed manner.

Collaborative optimization is appealing because the system-level optimizer or coordinator provides the disciplinary problems with targets for the shared variables and disciplinary outputs, while allowing each disciplinary optimization problem to manipulate its set of local design variables. In fact, if the targets were provided to the disciplines once and for all, complete autonomy would be attained. However, the system-level optimization is an iterative process and so the disciplinary subproblems continue to receive new targets, thus possibly necessitating human intervention when new outputs to the subsystems arrive with each system-level iteration.

8.3. Dimensionality of the system-level optimization problem. Because the disciplinary subproblems eliminate the local design variables from the system, both the system-level optimizer and the disciplinary optimization subproblems have a reduced number of variables, compared to the total set of design variables. However, this reduction is achieved at the price of system-level problems that are difficult or impossible to solve.

8.4. Ease of implementation and execution. When a problem formulation requires a multidisciplinary analysis capability, the effort is expended not just in implementing the MDA, but also during the execution of MDA, because the processing of the disciplinary inputs may require extensive human intervention. CO is claimed to ease problem implementation and execution because an explicit multidisciplinary analysis capability is not required. We believe this claim has not been proven satisfactorily as yet, as we now discuss.

On the positive side, local disciplinary variables need not be treated as optimization variables at the system level. On the other hand, although MDA is not implemented in CO, the flow of information among the disciplines still remains. That is, in our example of aero-structural interaction, structures still require input from aerodynamics, and conversely. This data exchange occupies a large portion of the implementation effort (regardless of the problem formulation) and adds to the execution effort. Because, typically, CO_2 would require many iterations to attain some level of convergence, human intervention to handle interdisciplinary inputs may be significant.

Moreover, because of the delinquent nature of the system-level problem in collaborative optimization, much time may be expended on fine-tuning both the problem formulation and the optimization algorithm in order to produce answers. And, as we have shown, these answers cannot be verified as being optimal or nearly optimal.

For small test problems, we have found that implementing CO was more laborious than implementing the standard method. This is precisely the case where small problems may not provide a fair test of the necessary implementation effort. We could not locate more than anecdotal information on the implementation effort in other publications on CO. This points to the need for careful measurement of labor expended on various parts of the problem implementation in MDO: the modeling, the problem statement, the optimization. Moreover, careful comparison must be made with the corresponding time expenditures when using, say, the standard formulation. Until such accounting is done on a realistic problem, substantiating claims on the increased ease of implementation will be difficult.

9. Concluding remarks. Bilevel approaches suggest themselves naturally as a decomposition strategy for large problems, and researchers have repeatedly turned to bilevel formulations in an attempt to deal with engineering systems in a decentralized fashion. While computational efficiency is one of the goals of bilevel approaches to the optimization of complex, coupled systems, the computational inefficiency that often results in practice is viewed, first, as a feature that will be ameliorated by increases in available computing power and, second, as less significant than the conjectured benefits of bilevel approaches, such as the ease of problem synthesis and implementation, disciplinary autonomy, or a problem decomposition that reflects certain organizational procedures.

The attempt to preserve disciplinary autonomy and reduce system-level complexity gives bilevel methods their intuitive appeal. However, to evaluate an approach to MDO, one must answer a number of questions concerning the resulting optimization problem(s). For methods based on decomposition and disciplinary autonomy, what manner of autonomy is actually afforded? What are the analytical and computational advantages and disadvantages attendant upon disciplinary autonomy? Do the benefits that motivate the use of disciplinary autonomy, such as ease of implementation or computational efficiency, actually obtain in the resulting approach?

In this work, we have examined in detail the analytical and computational features of a frequently proposed bilevel approach to MDO. The analytical features have a practical impact on the ability of nonlinear programming algorithms to solve the optimization problems that result from this approach. The study has illustrated the distinction between the intrinsic geometry of the feasible set and the way in which that set is represented in terms of constraints. Reformulated problems can introduce analytical features that cause difficulties for optimization algorithms [19, 21].

To illustrate the computational consequences of the analytical features of CO, we have conducted numerous tests on convex problems chosen for their simplicity in order to remove intrinsic difficulties of the test problems from the experiments. This means that the computational conditions of the tests were more benign than what can be realistically expected in practice. We also used analytical derivatives in the disciplinary subproblems and either highly accurate or analytical system-level sensitivities obtained from post-optimality sensitivity analysis of the disciplinary solutions. The numerical tests substantiated the analysis by revealing the following tendencies. Occasionally, a felicitous combination of optimization parameters and starting point would enable us to solve the CO system-level problem, although at considerably greater cost than the fully integrated approach. However, the solution could not be accomplished reliably. We observed that the solution was most reliably achieved when all the system-level iterates were strictly non-realizable (i.e., not realizable for *all* disciplines). This is a computational manifestation of the bad behavior of the system-level Jacobian at realizable values of the system-level variables (e.g., Propositions 6.2 and 6.4).

In this work, we have relied on SQP algorithms as the means of solving the CO system-level problem because SQP methods are generally the most efficient for equality constrained optimization. As we have shown, CO has analytical features that hinder the successful operation of SQP algorithms. Singularity of the system-level constraint Jacobian in CO_2 means that solutions of the CO system-level problem do not satisfy the standard Karush-Kuhn-Tucker optimality conditions, which leads to bogus estimates of Lagrange multipliers and penalty parameters. Moreover, an SQP method relies on projections into the nullspace of the constraint Jacobian. The Jacobian of the system-level constraints associated with a given discipline consists of the derivatives of the optimal value functions for disciplinary subproblems as functions of the system-level variables. Because the derivatives vanish at points realizable for that discipline, the Jacobian computation suffers from large numerical errors at or near realizable (and hence feasible) points. This makes projection into the nullspace of the system-level constraint Jacobians untenable and causes erratic computational behavior. Results for other classes of optimization algorithms applied to CO-like methods can be found in [36].

In summary, two characteristics of CO stand out. On the one hand, the approach does afford the user increased disciplinary autonomy. On the other, CO system-level problems are neither efficiently nor robustly solvable using conventional nonlinear programming algorithms. There is particular trouble at the points of interest, i.e., at or near realizable or interdisciplinary feasible points. While one can devise algorithmic approaches that avoid these points, one still must be sure that the ultimate solution is realizable, and hence physically meaningful, for all disciplines. In selecting a problem formulation, the user must weigh the relative importance of these features for the practical solution of the application problem in question.

Acknowledgments. The authors wish to thank Raphael Haftka for bringing to their attention the work in [25–27, 36], Stephen Robinson for a discussion of stability in nonlinear programming, and Sean Wakayama for the example that served as the basis for the one discussed in §4.1 Thanks are also due to Jean-François Barthelemy, Stephen Nash, and Thomas Zang for a careful reading of the draft and many helpful comments.

Appendix. To ease exposition in the main body of the paper, we considered an optimization problem in which only the disciplinary constraints were present. Here we present a complete problem notation for the case where no such assumption is made, because some problems demand the inclusion of system-level design constraints that are functions of outputs from both (or several) disciplines (see, e.g., [2]). We incorporate the alteration in Section 6.7 that insures that the disciplinary problems will always be feasible with respect to the disciplinary design constraints. The analysis presented in the paper applies to this general problem as well.

We thus consider the standard problem formulation

$$\begin{array}{ll} \underset{s,l_1,l_2}{\text{minimize}} & f(s, a_1(s,t_1,t_2), a_2(s,t_1,t_2)) \\ \text{subject to} & g_0(s, a_1(s,t_1,t_2), a_2(s,t_1,t_2)) \geq 0 \\ & g_1(s, l_1, a_1(s,t_1,t_2)) \geq 0 \\ & g_2(s, l_2, a_2(s,t_1,t_2)) \geq 0, \end{array} \tag{9.1}$$

where, given the shared and local design variables (s, l_1, l_2), the analysis outputs a_1, a_2 are the solution of the MDA system (2.1)–(2.2).

Reformulating (9.1) along the lines of CO, now requires the introduction of the third set of system-level constraints and a "Discipline 0" subproblem that represents the system-level design constraint. Consistency constraints that represent Disciplines 1 and 2 and the corresponding subproblems remain unchanged.

Optimization of "Discipline 0" treats the system-level design constraints to obtain $\bar{\sigma}_0(s,t_1,t_2)$, $\bar{\zeta}_1(s,t_1,t_2)$, and $\bar{\zeta}_2(s,t_1,t_2)$:

$$\begin{array}{ll} \underset{\sigma_0,\zeta_1,\zeta_2}{\text{minimize}} & \frac{1}{2}\left[\| \sigma_0 - s \|^2 + \| \zeta_1 - t_1 \|^2 + \| \zeta_2 - t_2 \|^2\right] \\ \text{subject to} & g_0(\sigma_0, \zeta_1, \zeta_2) \geq 0. \end{array} \tag{9.2}$$

Then for CO$_2$, the additional system-level consistency constraint is

$$c_0(s,t_1,t_2) = \frac{1}{2}\left[\| \bar{\sigma}_0(s,t_1,t_2) - s \|^2 + \| \bar{\zeta}_1(s,t_1,t_2) - t_1 \|^2 + \| \bar{\zeta}_2(s,t_1,t_2) - t_2 \|^2\right],$$

while for CO_1, the additional system-level consistency constraints are

$$\begin{aligned} c_0^1(s,t_1,t_2) &= \bar{\sigma}_0(s,t_1,t_2) - s \\ c_0^2(s,t_1,t_2) &= \bar{\zeta}_1(s,t_1,t_2) - t_1 \\ c_0^3(s,t_1,t_2) &= \bar{\zeta}_2(s,t_1,t_2) - t_2. \end{aligned}$$

REFERENCES

[1] H. M. Adelman, J. L. Walsh, and J. I. Pritchard, *Recent advances in integrated multidisciplinary optimization of rotorcraft, AIAA paper AIAA-92-4777-CP*, in Proceedings of the Fourth AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization, September 1992, pp. 710–721.

[2] N. M. Alexandrov and S. Kodiyalam, *Initial results of an MDO method evaluation study*, in Seventh AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, 1998. AIAA Paper 98-4884.

[3] N. M. Alexandrov and R. M. Lewis, *An analysis of some bilevel approaches to multidisciplinary design optimization.* Unpublished.

[4] ——, *Analytical and computational properties of distributed approaches to MDO.* In preparation for the 8th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization, September 2000.

[5] ——, *Comparative properties of collaborative optimization and other approaches to MDO*, in Engineering Design Optimization, MCB University Press, 1999.

[6] R. J. Balling and S. Sobieszczanski-Sobieski, *An algorithm for solving the system-level problem in multilevel optimization.* AIAA Paper 94-4333, September 1994. Presented at the Fifth AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City Beach, Florida, September 7–9, 1994.

[7] ——, *Optimization of coupled systems: A critical overview of approaches.* AIAA Paper 94-4330, September 1994. Presented at the Fifth AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City Beach, Florida, September 7–9, 1994.

[8] J.-F. M. Barthelemy, *Development of a multilevel optimization approach to the design of modern engineering systems.* NASA Contractor Report 172184, August 1983. Virginia Polytechnic Institute and State University, Blacksburg, VA.

[9] J. F. M. Barthelemy and M. F. Riley, *Improved multilevel optimization approach for the design of complex engineering systems*, AIAA Journal, 26 (1988), pp. 353–360.

[10] J. F. M. Barthelemy and J. Sobieszczanski-Sobieski, *Extrapolation of optimum design based on sensitivity derivatives*, AIAA Journal, 21 (1983), pp. 797–799.

[11] R. Braun, *Collaborative Optimization: An architecture for large-scale distributed design*, PhD thesis, Stanford University, May 1996. Department of Aeronautics and Astronautics.

[12] R. D. Braun and I. M. Kroo, *Development and application of the collaborative optimization architecture in a multidisciplinary design environment*, in Multidisciplinary Design Optimization: State of the Art, N. M. Alexandrov and M. Y. Hussaini, eds., SIAM, 1997, pp. 98–116.

[13] R. D. Braun, A. A. Moore, and I. M. Kroo, *Collaborative approach to launch vehicle design*, Journal of Spacecraft and Rockets, 34 (1997), pp. 478–486.

[14] F. H. Clarke, *Optimization and Nonsmooth Analysis*, Society for Industrial and Applied Mathematics,

Philadelphia, 1990.

[15] A. V. Fiacco, *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*, Academic Press, New York, 1983.

[16] A. V. Fiacco and J. Liu, *Degeneracy in NLP and the development of results motivated by its presence*, Annals of Operations Research, 46 (1993), pp. 61–80.

[17] A. V. Fiacco and G. P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, Classics in Applied Mathematics, SIAM, Philadelpha, 1990.

[18] J. P. Giesing and J. F. Barthelemy, *A summary of industry MDO applications and needs.* AIAA Paper 98-4737, 1998. 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Sept. 2–4, 1998, St. Louis, MO.

[19] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, *Software and its relationship to methods*, in Numerical Optimization 1984, P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., SIAM, Philadelphia, 1985, pp. 139–159.

[20] ——, *User's guide for NPSOL (Version 5.0): A Fortran package for nonlinear programming*, Stanford University, 1995.

[21] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, Academic Press, London, 1981.

[22] S. Kodiyalam, *Evaluation of methods for multidisciplinary design optimization, Phase I*, Tech. Rep. NASA/CR-1998-208716, NASA Langley Research Center, September 1998.

[23] I. M. Kroo, *MDO for large-scale design*, in Multidisciplinary Design Optimization: State of the Art, N. M. Alexandrov and M. Y. Hussaini, eds., SIAM, 1997, pp. 22–44.

[24] V. M. Manning, *High speed civil transport design using cololaborative optimization and approximate models.* A final report submitted to NASA Langley Research Center in partial fulfillment of the research funded by Grant NCC-1-253, 1999.

[25] L. A. Schmit and K. J. Chang, *A multilevel method for structural synthesis*, in A collection of technical papers : AIAA/ASME/ASCE/AHS 25th Structures, Structural Dynamics and Materials Conference, American Institute of Aeronautics and Astronautics, 1984.

[26] L. A. Schmit, Jr. and M. Mehrinfar, *Multilevel optimum design of structures with fiber-composite stiffened-panel components*, AIAA Journal, 20 (1982), pp. 138–147.

[27] L. A. Schmit, Jr. and R. K. Ramanathan, *Multilevel approach to minimum weight design including buckling constraints*, AIAA Journal, 16 (1978), pp. 97–104.

[28] I. Sobieski and I. Kroo, *Aircraft design using collaborative optimization.* AIAA paper 96-0715, presented at the 34th AIAA Aerospace Sciences Meeting, Reno, Nevada, Jan. 15-18, 1996, 1996.

[29] I. P. Sobieski, *Collaborative Optimization Using Response Surface Estimation*, PhD thesis, Stanford University, 1998.

[30] I. P. Sobieski, V. M. Manning, and I. M. Kroo, *Response surface estimation and refinement in collaborative optimization*, 1998. AIAA Paper 98-4753.

[31] J. Sobieszczanski-Sobieski, *A linear decomposition method for large optimization problems—blueprint for development*, Tech. Rep. TM 83248, NASA, 1982.

[32] ——, *Optimization by decomposition: A step from hierarchic to non-hierarchic systems*, Tech. Rep. TM 101494, NASA, Hampton, Virginia, September 1988.

[33] ——, *Two alternative ways for solving the coordination problem in multilevel optimization*, Structural Optimization, 6 (1993), pp. 205–215.

[34] J. Sobieszczanski-Sobieski, B. James, and A. Dovi, *Structural optimization by multilevel decomposition*, in Proceedings of the AIAA/ASME/ASCE/AHS 24th Structures, Structural Dynamics, and Materials Conference, May 1983. AIAA Paper 83-0832-CP.

[35] J. Sobieszczanski-Sobieski, B. B. James, and M. F. Riley, *Structural sizing by generalized, multilevel optimization*, AIAA Journal, 25 (1987), pp. 139–145.

[36] R. Thareja and R. T. Haftka, *Numerical difficulties associated with using equality constraints to achieve multi-level decomposition in structural optimization.* AIAA Paper 86-0854, 1986.

[37] J. Walsh, K. Young, J. Pritchard, H. Adelman, and W. Mantay, *Integrated aerodynamic/dynamic/structural optimization of helicopter rotor blades using multilevel decomposition*, Tech. Rep. NASA TP 3465, NASA Langley Research Center, January 1995.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. AGENCY USE ONLY ***(Leave blank)***

2. REPORT DATE
April 2000

3. REPORT TYPE AND DATES COVERED
Technical Memorandum

4. TITLE AND SUBTITLE
Analytical and Computational Aspects of Collaborative Optimization

5. FUNDING NUMBERS
522-31-21-08

6. AUTHOR(S)
Natalia M. Alexandrov and Robert Michael Lewis (ICASE)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681–2199

8. PERFORMING ORGANIZATION REPORT NUMBER
L–17988

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA/TM–2000–210104

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified-Unlimited
Subject Category 64 Distribution: Standard
Availability: NASA CASI (301) 621–0390

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

Bilevel problem formulations have received considerable attention as an approach to multidisciplinary optimization in engineering. We examine the analytical and computational properties of one such approach, collaborative optimization. The resulting system-level optimization problems suffer from inherent computational difficulties due to the bilevel nature of the method. Most notably, it is impossible to characterize and hence identify solutions of the system-level problems because the standard first-order conditions for solutions of constrained optimization problems do not hold. The analytical features of the system-level problem make it difficult to apply conventional nonlinear programming algorithms. Simple examples illustrate the analysis and the algorithmic consequences for optimization methods. We conclude with additional observations on the practical implications of the analytical and computational properties of collaborative optimization.

14. SUBJECT TERMS
Bilevel optimization, collaborative optimization, constraint qualification, decomposition, multidisciplinary design optimization, multilevel optimization, nonlinear programming, optimality conditions

15. NUMBER OF PAGES
31

16. PRICE CODE
A03

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT